RADC-TR-89-358
In-House Report
December 1989

ADAPTIVE RESONANCE THEORY I

Scott S. Shyne

Rome Air Development Center
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE — *Form Approved OMB No. 0704-0188*

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS N/A
2a. SECURITY CLASSIFICATION AUTHORITY N/A 2b. DECLASSIFICATION / DOWNGRADING SCHEDULE N/A	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
4. PERFORMING ORGANIZATION REPORT NUMBER(S) RADC-TR-89-358	5. MONITORING ORGANIZATION REPORT NUMBER(S) N/A
6a. NAME OF PERFORMING ORGANIZATION Rome Air Development Center 6b. OFFICE SYMBOL *(If applicable)* IRRA	7a. NAME OF MONITORING ORGANIZATION Rome Air Development Center (IRRA)
6c. ADDRESS *(City, State, and ZIP Code)* Griffiss AFB NY 13441-5700	7b. ADDRESS *(City, State, and ZIP Code)* Griffiss AFB NY 13441-5700
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Rome Air Development Center 8b. OFFICE SYMBOL *(If applicable)* IRRA	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N/A
8c. ADDRESS *(City, State, and ZIP Code)* Griffiss AFB NY 13441-5700	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 62702F · PROJECT NO. 4594 · TASK NO 18 · WORK UNIT ACCESSION NO. N3

11. TITLE *(Include Security Classification)*
Adaptive Resonance Theory I

12. PERSONAL AUTHOR(S)
Scott S. Shyne

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
In-House	FROM Jun 89 TO Aug 89	December 1989	32

16. SUPPLEMENTARY NOTATION
N/A

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
12	01	
17	08	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Adaptive Filters · Unsupervised Learning · Pattern Classification · Neural Network

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

This report describes the function, operation, test and evaluation of a Neural Network that accomplishes unsupervised learning of binary input patterns by classifying them using Adaptive Resonance Theory. [illegible]

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL MICHAEL L. HINMAN	22b. TELEPHONE *(Include Area Code)* (315) 330-7788 22c. OFFICE SYMBOL RADC(IRRA)

DD Form 1473, JUN 86 — *Previous editions are obsolete.*

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

Table of Contents

I. Introduction to Neural Networks

Neural Networks are an information processing technology that were first introduced in the early 1940's by McCulloch and Pitts. They published a paper that compared the human brain to a computer. Their paper stated that the internal architecture of the brain was composed of billions of interconnected neurons, where a neuron was described as a single well-defined computing element. The threshold-logic view that McCulloch and Pitts held of the basic operations was not quite accurate. Nonetheless, this must be seen as one of the initial attempts to theoretically model the activity of the brain.

In 1954, Farley and Clark simulated stimulus-response relationships in random networks of learning elements. This idea was further elaborated circa 1960 when more specific functions and networks were created by Rosenblatt, Widrow, Hoff, Steinbuch, Caianiello and some others. Neural Network technology had been progressing steadily through the 50's and mid 60's when in 1969, Minsky and Pappert wrote a book, "Perceptrons", that contained a mathematical analysis of the computing powers of perceptrons, with an emphasis on their limitations. A single layer perceptron is a simple network that can be used to learn and recognize simple patterns. The perceptron decides whether an input belongs to one of two classes based on a weighted sum of the input elements minus a threshold. The resulting activation energy is passed through a hard limiting nonlinearity such that the output of the node is either +1 or -1. Minsky and Pappert, concluded that the Neural Network model for a single layer perceptron was incapable of learning the exclusive-OR (XOR) problem and was therefore only capable of solving trivial problems. This book successfully set research back by about 15 years. Dedicated Neural Network researchers continued working on certain aspects of the technology but the spotlight of scientific research was otherwise directed. It was not until the late 70's and early 80's that Neural Networks began to come back into the scientific mainstream. Multilayer perceptrons were used to solve the exclusive-OR problem thereby generating interest in Neural Networks again. The technology of neural computing has been gaining popularity and visibility since that time.

Neural Network architectures offer many advantages over standard VonNeumann architectures. Because there are more processing nodes, Neural Networks are good for solving unstructured problems that require a large number of constraints to be satisfied. Damage to a few connections or neurons does not necessarily impair the overall performance of the system. Most neural nets also adapt connection weights in time to improve performance based on current results. This characteristic is known as feedback. Adaptation (feedback) provides a degree of robustness by compensating for minor variations in processing elements.

Strong application environments for Neural Networks include the areas of pattern classification, pattern reconstruction and optimal path determination. These types of problems require a system to consider many separate factors before following a specific course of action. Variables such as the size of a given pattern, the sparsity of input in that pattern and the type of input (binary or continuous) must be considered every time a system is run on new data. The system must be able to learn based on its input while still maintaining the integrity of previous information. Some of the pattern classification problems that have been researched using neural networks include differentiating similar sounds, classifying the propeller noise of different ships and experiments in speech and character recognition.

II. Background Information on Adaptive Resonance Theory (ART)

Stephen Grossberg is widely accepted as being responsible for performing extensive work relating to the development of Adaptive Resonance Theory. Grossberg completed his senior fellowship thesis in 1961 at Dartmouth College. At least thirty years of his life have been spent trying to understand and recreate certain thought processes of the human brain. Stephen Grossberg's research has produced at least forty technical papers developing the basic ideas of Adaptive Resonance Theory. Most of his works relate to simulating memory activity through the use of applied mathematics and computer science.

Adaptive Resonance Theory has the ability to self-organize and self-stabilize recognition code in response to an arbitrary list of input patterns. It uses a type of adaptive filter to strain out noise from an input pattern. ART has two basic configurations. The first algorithm (ART1) uses unsupervised learning techniques on binary input patterns. Bidirectional connections exist between the input layer and the output layer. Lateral inhibition exists between all of the output nodes. Pure mathematicians like this network because it can be completely described using nonlinear differential equations. The differential equations can be shown to implement the sequential leader clustering algorithm. The second algorithm (ART2) also uses unsupervised learning techniques but the input is continuous instead of binary. This paper will be restricted to studying the aspects of ART1. For further interest in (ART2) please refer to [1] Carpenter, G. and Grossberg, S. 1987 and [2] Carpenter, G. and Grossberg, S. 1989.

III. Scope

The specific focus of this paper is to discuss the ART1 algorithm and provide an evaluation of a simulation of ART1 that was written and tested at Rome Air Development Center (RADC). Memory consolidation and noise tolerance are two important aspects that are discussed in the evaluation. The results of this simulation are discussed in the test and evaluate section of this paper.

IV. Technical Description

A. Hardware and Software Description

The software that implements the ART1 algorithm was created at the Rome Air Development Center, Intelligence and Reconnaissance Directorate, Griffiss Air Force Base, NY. The program was written in Borland's Turbo C version 2.0. The program runs on Advanced Technology (AT) compatible WYSEpc 286 terminals. The monitor is a Tatung CM-1380F EGA type color CRT. The software can run from one 5.25 inch floppy. The program can be loaded to a hard drive providing all necessary files are copied from the original floppy disk. The software development was accomplished using a structured programming methodology whereby development began at a high level of abstraction from algorithmic methods and was refined by repeatedly increasing the level of detail to produce standardized code. Modularity was introduced to the program through the use of include files and functional definitions called by the program. The graphic display is only portable given Borland's Turbo C environment and associated AT compatible computers.

B. Theoretical Architecture

As mentioned earlier, ART1 is a bidirectional Neural Network consisting of an input layer and an output layer. This means that each input node is connected to each output node and there is a separate connection from each output node to each input node. Lateral inhibition exists between the output nodes although there are no physical connections. The conjectural connection between the output nodes is accomplished by applying the max function to the activation energy of each output node. The exemplar with the maximum activation energy is selected to undergc the vigilance test. If the exemplar passes the vigilance test, the chosen exemplar becomes the best matching exemplar. If the exemplar does not pass the vigilance test, it is temporarily disabled to allow the search for a best matching exemplar to continue. This two layer network contains no hidden layers. When using ART1 in a pattern recognition scenario, the number of input nodes is generally equal to the number of output nodes.

C. Mathematical Viewpoint

The equations of ART1 implement the simple sequential leader clustering algorithm. Procedures can be concisely stated using relatively simple formulas and standard Neural Network terminology. The initialization procedure assigns a value of 1 to all the top-down connections. The bottom up connections are given the value of 1 divided by the number of nodes plus one. Both the top-down and bottom-up procedures are initialized at time = 0. This can be succinctly stated as follows :

Top-down connection	Bottom-up connection
$t_{ij}(0) = 1.0$	$b_{ij}(0) = \frac{1}{1 + n}$
$0 <= i <= n - 1,$	
$0 <= j <= m - 1$	set p to $0 <= p <= 1$

The next procedure computes the matching scores for each exemplar based on the activation energy derived from the input value and the bottom up connection. In the following equation, u_j is the output of output node j and x_i is the ith element of the input having a value of 0 or 1. The letter t is the time frame at which the activation energy is formulated.

$$u_j = \sum_{i=0}^{n-1} b_{ij}(t) * x_i$$

Lateral inhibition is achieved by selecting the exemplar with the maximum activation energy. The exemplar chosen is known as the best matching exemplar. It is used to determine which exemplar is chosen to represent a particular cluster of patterns.

$$u_j^* = \overset{\max}{j} \{u_j\}$$

The vigilance test determines the distance that the input pattern is from the current best matching exemplar. If the pattern is close enough, the input pattern and the cluster are combined to form a currently updated cluster.

$$\text{sum\_of\_inputs} = \sum_{i=0}^{n-1} x_i$$

$$\text{sum\_of\_chosen} = \sum_{i=0}^{n-1} t_{ij} * x_i$$

The last procedure simulates the computer's ability to learn. Once a best matching exemplar has been found and it is within the specified vigilance, the appropriate top-down and bottom-up connections are modified based on the input data. This modification allows the network to remember what the current pattern looks like and what cluster it is stored in. When adapting connection weights, both top-down and bottom-up connections are increased when the input node receives a one. If an input node receives a zero, both sets of connection weights are decreased or weakened. The following equations show how to redefine the connections based on the input values:

New top-down connection :

$$t_{ij}(t + 1) = t_{ij}(t) * x_i$$

New bottom-up connection :

$$b_{ij}(t+1) = \frac{t_{ij}(t) * x_i}{.5 + \sum_{i=0}^{n-1} t_{ij}(t) * x_i}$$

D. Simulation Description

The algorithm introduced by Gail Carpenter and Stephen Grossberg is appropriately called the Carpenter/Grossberg classifier. It performs the classification of patterns using a similar technique to the sequential leader clustering algorithm. This clustering algorithm forms clusters of different patterns one after the other. Each output node contained in the network has the ability to define a separate cluster. When a new pattern is read as input, the cluster with the highest activation energy is compared to the input pattern. If the input pattern fits in the cluster with the highest energy, the cluster is redefined by logically ANDing itself with the input pattern. The redefinition of the cluster also becomes the output of the network. If the pattern does not fit in the chosen cluster, that cluster is temporarily ignored and the cluster with the next highest activation energy is compared to the input. This process continues until the pattern matches a cluster or none of the defined clusters match the input pattern. In the latter case, a new cluster is defined where the next unused exemplar represents the new pattern. The connection weights are adjusted accordingly. When all of the output nodes define a cluster, no more distinctive patterns can be added.

The distance between the clusters is determined by the vigilance threshold. The vigilance threshold is a floating point number between zero and one that the user enters at the beginning of the program. The greater the vigilance threshold, the greater the distance between the clusters. When the threshold is high, the clusters are sharply defined. In order for a match to occur, the pattern must be very close to the stored cluster. If the vigilance is decreased slightly, the input pattern does not have to be an exact match with the cluster. The lowering of the vigilance threshold has a clouding or blurring effect that permits closely related patterns to be matched with the same cluster. This aspect of ART1 allows the designer to introduce a certain amount of noise tolerance into the system. By varying the vigilance threshold, noisy patterns can be matched to their correct clusters. Noise is introduced to an unblemished pattern by randomly flipping each pixel (probability defined by the user). The noisy pattern can now be put through the Carpenter/Grossberg classifier in an attempt to recognize the pattern by matching it with a predefined cluster. When testing noisy patterns on a previously trained network, no new clusters are added and no predefined clusters are modified. Learning can be continued even after training by removing the comments around the call to adapt_best_exemplar procedure in the testing portion main program. Each noisy input pattern will have some redefining effect on the stored cluster.

V. Test and Evaluate

The program is flexible enough to handle differently configured input files. The size of the input file must be known prior to executing the program. The input patterns used for the testing are square in configuration but this is not a requirement. The program was tested on 5x5, 6x6, 7x7, 8x8, 9x9, and 10x10 size input patterns. Each input file contained six patterns. The time that the program took to run increased on the order of n^3.

Table I

Size	Time (sec)
5x5	4.01
6x6	6.11
7x7	9.98
8x8	16.07
9x9	24.21
10x10	34.34

The program was tested on two separate types of files. The complete set of input data is listed at the end of the paper under alphabet data file and weapons data file. The first file contained 26 different patterns. Each pattern was a 7x7 binary representation of a letter of the English alphabet. When a vigilance threshold of 0.9 was used, each of the 26 patterns defined a separate cluster. If the vigilance was lowered to 0.8, several of the patterns were stored in the same matching cluster. With 26 separate clusters and a vigilance threshold of 0.9, the simulation was not very noise tolerant. Seven percent noise seemed to be the maximum acceptable for correct classification of a pattern to occur. The following examples show how ART1 performs in a slightly noisy environment :

Example 1 :

```
   ORIGINAL              DEGRADED              OUTPUT

    ****                  ****                  ****
    *  *                     *                     *
    *  *                  *  *                  *  *
    ****                  ****                  ****
    *  *                  *  *                  *  *
    *  *                  *  *                  *  *
    ****                  **** *                ****
```

Image Degraded by 4%
Matched Pattern 2

Example 2 :

```
   ORIGINAL              DEGRADED              OUTPUT

   ******                ******                ******
   *                     *   *                 *
   *                     *                     *
   *****                 **  *                 **  *
   *                     *                     *
   *                     *                     *
   *                     *                     *
```

Image Degraded by 6%
Matched Pattern 6

The previous two examples illustrate the adaptive filtering mechanism that classifies the degraded patterns. If a pixel is degraded within a pattern, the output pattern still maintains that degraded pixel. If a pixel outside the pattern is degraded, that noise is eliminated by the classifier. If a pattern does not match a previously learned pattern, a new cluster is formed. This creation of a new cluster is due to the fact that too much noise has been introduced and the pattern was not recognized. Example 3 illustrates this idea:

Example 3 :

```
   ORIGINAL              DEGRADED              OUTPUT

     ***                 **** *                **** *
    *   *                *   *                 *   *
    *   *                *   *                 *   *
    *   *                *   *                 *   *
    * * *                * * *                 * * *
     ***                    *                     *
        *                    *                     *
```

Image Degraded by 8%
Matched Pattern 27

The degradation was too much for the classifier to correctly recognize so a new cluster, pattern 27, was formed. The important aspect to remember is that this network performs very well under perfect input. Adaptive Resonance Theory excels in the area of memory consolidation. A 7x7 pattern has 49 input nodes and 49 output nodes. With a total of 98 nodes, the network has the ability to learn and remember 49 distinct patterns (same as the number of output nodes). For the sake of comparison, most Hopfield Networks are very noise tolerant but they can only remember a small number of patterns. The number of patterns that a Hopfield Network can remember is usually no more than fifteen percent of the total number of output nodes. The number of patterns that ART1 remembers is a maximum of fifty percent of the total number of nodes.

The second type of file that the network was tested on consisted of six patterns. Each pattern was a 7x7 binary (0 or 1) matrix in the form of different mission targets. The network is trained by reading the six perfect input patterns and forming six separate clusters. The vigilance threshold is set at .75 so that an exact match is not necessary for correct classification. These clusters become the basis for the simulation. The adapt_best_match procedure is disabled once training has been completed so that the correctly defined clusters are not modified by noisy input data. This allows the user the opportunity to see how ART1 performs with the introduction of noise.

Example 4 :

```
ORIGINAL        DEGRADED        OUTPUT

  * *             * *             * *
   *               *               *
*  *  *         * **  *         *  *  *
 *****           ****            ****
  ***             ***             ***
   *            *  * *             *
   *               *               *
```

Image Degraded by 8%
Matched Pattern 1

The image in example 4 was degraded by the same percentage as the image in example 3. The reason example 4 was recognized and example 3 was not recognized is because the vigilance threshold was lowered from .9 to .75. By lowering the vigilance threshold, images that are not exactly alike can be matched to the same cluster. This allows the system to be more noise tolerant.

Example 5 :

ORIGINAL

```
   *
   *
  ***
 *****
*******
   *
  * *
```

DEGRADED

```
*  *  *
   *
  *** *
  ***
* ** **
   * *
  * *
```

OUTPUT

```
   *
   *
  ***
  ***
* ** **
   *
  * *
```

Image Degraded by 16%
Matched Pattern 2

Example 6 :

ORIGINAL

```
  **
 ****
  **
 ****
 ****
 ****
 ****
```

DEGRADED

```
* ** *
 **
   *  *
 ** *
 ******
 *****
 ****
```

OUTPUT

```
  **
 **
   *
 ** *
 ****
 ****
 ****
```

Image Degraded by 20%
Matched Pattern 3

The following table lists the average maximum amount of noise over 100 sample runs that could be introduced for each of the six test patterns:

Table II

Pattern	Avg Maximum Noise %
B52	.10
F15	.15
APC	.20
Truck	.15
Tank	.20
Helicopter	.15

On some patterns, a degradation of up to twenty percent could be correctly classified. The average maximum noise varies from one pattern to the next due to the sparsity of the input pattern (how many pixels are activated and how close they are to each other). Once the network had been trained on perfect data, the clusters were no longer modified by the input patterns. The network can continue its learning process by enabling the adapt_best_exemplar procedure during the testing phase of the simulation. This will allow the program to modify each cluster based on the noisy input and if the noisy input is not matched to any other cluster, a new cluster is created at the max exemplar.

VI. Recommendations

Adaptive Resonance Theory exhibits strengths in the area of memory consolidation since many patterns can be stored in a relatively small amount of memory. By adjusting the vigilance threshold and decreasing the number of patterns, some degree of noise tolerance can be introduced.

Further development of ART1 is necessary to better improve the algorithm's performance when a larger amount of noise is introduced. There are several possible modifications that may significantly increase ART1's efficiency and reliability in a degraded environment. After the network has been trained on a set number of patterns, the vigilance threshold might need to be adjusted. If the original training clusters are distinct enough, a larger percentage of noise should be tolerated in the system. This is due to the fact that the more distinct the patterns are, the greater the distance between clusters. A second method that may increase reliability is to adapt the connection weights more slowly. This will make the network adapt more slowly thereby maintaining the original training structure for a longer period of time.

Possible applications for the ART1 learning algorithm might include handwritten letter recognition. Systems already exist that have the ability to recognize handwritten letters within a noise degradation of ten percent. The increase in noise tolerance would be a valuable contribution to Adaptive Resonance Theory research.

Another possible application is in the area of front-end adaptive filtering. ART1 could be implemented as one part of a complex hybrid network. If several networks are layered, the strong points of all algorithms involved may be demonstrated. ART1 can classify a very noisy pattern into a very clouded image and then another algorithm,possibly a Hopfield Network,can sharpen the image to original or close to original form. This layering of networks has the potential to produce the best reconstructed image of any other network.

VII. Conclusion

Neural Networks offer a powerful technique for solving artificial intelligence problems. The distributed parallel architecture offers faster execution time than standard AI problem solving algorithms provided inherent parallelism is exploited. The ability to learn without extensive knowledge of an environment is an aspect of Neural Networks that is not present in any other area of computer science. Adaptive Resonance Theory uses adaptive filtering to classify patterns into related clusters. The ability to remember a large number of patterns while consuming only a small amount of memory is a characteristic of ART1 that sets it apart from other Neural Network algorithms. Noise tolerance is an important aspect to consider in the area of pattern recognition. The Hopfield Network is generally accepted as being one of the most effective reconstruction networks for restoring degraded patterns to their original configuration. ART1 has the potential to increase the efficiency of a Hopfield Network by acting as an adaptive filter at the front end of the network. ART1 classifies the pattern into a clouded but recognizable shape and the Hopfield Network sharpens the picture by rebuilding the original pattern. These ideas still require much development but the potential results are inspiring.

BIBLIOGRAPHY

1.) Carpenter, G.A. and Grossberg, S., ART2 : Self-Organization of stable category recognition codes for analog input patterns. Applied Optics : special issue on neural networks, 1987.

2.) Carpenter, G.A. and Grossberg, S., Invariant Recognition of Cluttered Scenes by a Self-Organizing ART Architecture : CORT-X Boundary Segmentation. Neural Networks, Vol2, pp. 169-181, 1989.

3.) Caudill, Maureen, Neural Networks Primer part V. AI Expert, November 1988.

4.) Grossberg, S., Neural Networks and Natural Intelligence. Cambridge, Ma., MIT press, 1988.

5.) Klimasauskas, Casmir C. "Casey", Neural Networks : A Short Course From Theory to Application. PC AI November/December 1988.

6.) Kohonen, Teuvo, Adaptive, Associative, and Self-Organizing functions in Neural Computing. Applied Optics, Vol. 26, no. 23, December 1987.

7.) Lippmann, Richard P., An Introduction to Computing with Neural Networks. IEEE ASSP Magazine, April 1987.

Weapons Data File
------- ---- ----

```
  * *
   *
*  *  *
 *****
  ***
   *
   *
```

Pattern 1
B52

```
   *
   *
  ***
 *****
*******
   *
  * *
```

Pattern 2
F15

```
 **
****
 **
****
****
****
****
```

Pattern 3
APC

```
 ******
 ******
*******
 *   *
```

Pattern 4
Truck

```
   **
******
   ***
*******
*******
```

Pattern 5
Tank

```
*     *
 * * *
  ***
  ***
 * * *
*  *  *
  **
```

Pattern 6
Helicopter

Alphabet Data file

```
  *
 * *
*   *
*****
*   *
*   *
*   *
```
Pattern 1

```
****
*   *
*   *
****
*   *
*   *
****
```
Pattern 2

```
 *****
*
*
*
*
*
 *****
```
Pattern 3

```
****
*   *
*   *
*   *
*   *
*   *
****
```
Pattern 4

```
******
*
*
****
*
*
******
```
Pattern 5

```
******
*
*
*****
*
*
*
```
Pattern 6

```
 *****
*
*
* ***
*    *
*    *
 ****
```
Pattern 7

```
*     *
*     *
*     *
*******
*     *
*     *
*     *
```
Pattern 8

```
*******
   *
   *
   *
   *
   *
*******
```
Pattern 9

```
 *****
   *
   *
   *
   *
*  *
 **
```
Pattern 10

```
*   *
*   *
*  *
***
*  *
*   *
*   *
```
Pattern 11

```
*
*
*
*
*
*
*****
```
Pattern 12

```
**   **
* * * *
*  *  *
*     *
*     *
*     *
*     *
```
Pattern 13

```
*     *
**    *
* *   *
*  *  *
*   * *
*    **
*     *
```
Pattern 14

```
 ****
*    *
*    *
*    *
*    *
*    *
 ****
```
Pattern 15

```
 ****
*    *
*    *
*****
*
*
*
```

Pattern 16

```
 ***
*   *
*   *
*   *
* * *
 ***
    *
```

Pattern 17

```
*****
*    *
*    *
*****
*    *
*    *
*    *
```

Pattern 18

```
  ***
 *   *
 *
  ***
     *
*    *
 ****
```

Pattern 19

```
*******
   *
   *
   *
   *
   *
```

Pattern 20

```
*   *
*   *
*   *
*   *
*   *
 ***
```

Pattern 21

```
*     *
 *   *
 *   *
 *   *
  * *
  * *
   *
```

Pattern 22

```
*     *
*     *
*     *
*  *  *
* * * *
**   **
*     *
```

Pattern 23

```
*     *
 *   *
  * *
   *
  * *
 *   *
*     *
```

Pattern 24

```
*     *
 *   *
  * *
   *
   *
   *
   *
```

Pattern 25

```
*******
     *
    *
   *
  *
 *
*******
```

Pattern 26